Scalable and Robust Coordination of Multiple Mobile Robots using PROSA and delegate MAS

Johan Philips[i] Paul Valckenaers Hendrik Van Brussel Herman Bruyninckx

Department of Mechanical Engineering,
KU Leuven
Heverlee, Belgium
e-mail: {firstname.lastname}@mech.kuleuven.be

***Abstract*— In manufacturing control, the ideas of explicit resource allocation and intention propagation proved to increase scalability and robustness of the overall system. Moreover, making the environment a first class citizen in the underlying software system, results in a higher decoupling. This paper investigates the applicability of these concepts in robotics. More specifically, to illustrate the benefits a multi-robot scenario is discussed in which a set of robots navigate through a restricted environment. The need for coordination in such an environment is shown and distributing this coordination through environment resources enhances scalability and system robustness with respect to uncertainty and disturbances.**

Keywords: multi-robot coordination, navigation, mobile robots.

I. Introduction

Currently, most robot software architectures follow a *robocentric* approach which lacks a proper reflection of reality and hinders the creation of scalable and robust robot software with respect to external disturbances and changes in the world-of-interest. The key problem is the implicit resource allocation inherent in their design. In multi-robot applications in dynamic environments, e.g. a fleet of AGVs maneuvering independently from each other in a shared warehouse, actions taken by one robot can have consequences for other robots or humans. Therefore, coordination is required to minimize the interference between the various entities. For instance, driving into a narrow corridor might block other users of that corridor even if they had entered it first.

In manufacturing control, the paradigm shift toward explicit resource awareness and allocation has led to insights into improving scalability and robustness. The key idea is to explicitly represent the environment as resources, which require allocation.

The Product-Resource-Order-Staff Architecture (PROSA) and short-term forecasting mechanism, called 'delegate MAS' (D-MAS) [1], have successfully been applied in numerous applications in the past in several industrial settings [2], such as manufacturing control (large car body paint shop with various manufacturing resources arranged in a complex topology), open-air engineering (agriculture harvesting) and logistics (cross-docking).

This paper applies the same insights to the domain of robotics and more concretely to multi-robot navigation coordination in dynamic environments. In the test setup the coordinated approach was compared to a reactive approach and showed gain with respect to travelled distance and deviation from the planned trajectory. Moreover, PROSA and D-MAS were seamlessly integrated with existing planning and navigation software running on the robots.

II. Approach

The application discussed in this paper consists of a set of robots navigating independently from each other in the same environment. Their autonomous navigation should be smooth and interference with other robots or humans, also present in the environment, should be minimized.

A. Applicability

In our test setup we used robotic wheelchairs available in our lab. These wheelchairs could be used in a retirement home or hospital where a limited number of robotic wheelchairs should provide autonomous navigation to a larger group of patients or inhabitants. At the user's request these robots would navigate autonomously to the user. After the user is assisted into the wheelchair, the robot navigates to a given target location. The benefit of such an approach is that medical staff is only required when the user wants to mount or dismount the wheelchair. The robotic wheelchairs are able to avoid obstacles using range sensors. In this scenario the need for smooth navigation and low interference is apparent. Minimizing the patient's discomfort is a key criterion in this scenario.

Another possible scenario, in which the coordination of a fleet of autonomous robots is beneficial, is a factory or warehouse where a set of autonomously guided vehicles (AGVs) transport goods from one location to another. Also here, these AGVs would maneuver in an environment shared with humans and not have to follow a fixed trajectory and interference should be mimimal.

In Fig. 1 the need for coordination in such a multi-robot setup is depicted. In both environments either a *livelock* situation or suboptimal path execution can occur. In a livelock two or more entities constantly change their behavior with regard to one another, none of them progressing in their task. This is caused by the *robocentric* approach most current robot application follow. Each robot executes its own task, assuming the environment is implicitly allocated for its needs and not being aware other users might be present. Users can either be humans moving about or also other robots executing a task.

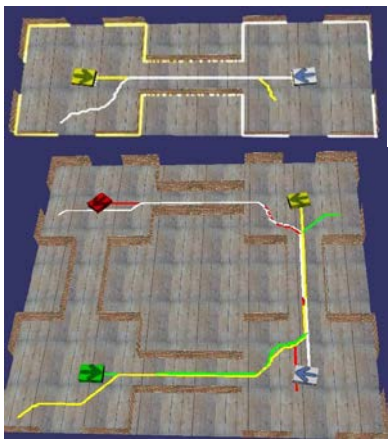

Fig. 1. Environments leading to livelocks or suboptimal path execution in a robocentric approach.

In the top environment only one corridor connects the rooms and it is too narrow for two robots to cross each other. Obviously, when both robots enter the corridor they will block each other at some time resulting in a livelock. On the other hand, in the bottom environment multiple paths to a target location are possible. Each room is connected to two narrow corridors and depending on the target going through one corridor is more efficient than going through the other. This scenario can also cause livelocks, provided the number of robots is high, but a more common problem is to execute an optimal path from one room to another. If two robots enter the same corridor and thus are not aware of each other's intention, they will most likely replan their route through the other corridors to still reach their target. This results in unnecessary travel.

III. Related Work

Typically in multi-robot navigation, a multi-robot path planning approach is used to ensure no collisions or livelocks occur. Multi-robot path planning can be split up in two categories: *centralized* and *decoupled* planning.

In centralized multi-robot path planning configuration spaces of individual robots are combined to one configuration space. The size of such a combined space increases exponentially with the number of robots, therefore most practical implementations use heuristics to narrow down the search space.

Decoupled multi-robot path planning on the other hand calculates first the individual paths of each robot and then tries to combine them to resolve conflicts. Commonly, priorities are assigned to robots and following this ranking paths for the robots are planned. Lower priority robots then take into account the already planned paths of higher priority robots. Although this solution is computationally efficient, it does not provide an optimal solution and deadlocks can occur when a higher priority robot blocks a lower priority robot.

Another commonly used decoupled planning technique is the path coordination method in which each robot is constraint to one or more paths, referred to as fixed-path and fixed-roadmap coordination. Some literature on multi-robot path planning, such as [3,4], provides scalable solutions.

A more recent approach uses subdimensional expansion in which not the full configuration space but a low dimensional subspace is considered [5]. It assumes the set of robots are loosely coupled, i.e. planning in a joint configuration space in not necessary. Indeed, in most cases, the robots are spread around in the environment. Planning is then first done for each individual robot and the dimensionality of search space is locally augmented if collisions occur.

Nevertheless, these solutions usually utilize a central planner with complete knowledge of the environment and all robots moving in it. Such a *closed world assumption* limits system extendibility. Instead, decomposing the planning over the entities in the environment (e.g. rooms, corridors) results in a scalable solution in which only relevant local knowledge (e.g. which robot will pass in this room at what time) is kept at the relevant entities.

Research in multi-robot coordination is generally also divided in two categories based on *where* this coordination is being done: *centralized* at one entity or *distributed* among several agents.

In a centralized approach the multiple robots are coordinated by a central computer or one of the robots is assigned as team leader. All degrees of freedom are then combined to offer an optimal solution to the specified task. See e.g. [6]. A centralized system suffers from most other problems a centralized coordination approach suffers from. Most of which are similar problems as the ones a centralized path planning approach encounters:

Firstly, this approach becomes computationally difficult as complexity is exponential with respect to the number of robots involved.

Secondly, it assumes all sensor and state information of all robots can be collected by the single coordinator and that this information does not change during planning. In unknown or dynamic environments, where communication is limited, these assumptions are unrealistic.

Thirdly, if the leader or coordinator malfunctions, a new leader must be available or the entire task fails.

Distributed coordination addresses the above described problems. Each robot acts on locally available information. Coordination is distributed among all robots and, for instance, a robot can coordinate with other robots nearby to solve a sub-task it cannot solve alone.

Typically, computational complexity is alleviated since each robot plans its own activities. Moreover, communication is also limited and local and robots are able to respond to unknown or dynamic environments better.

The downside of this approach is the inability to provide optimal solutions for the global task. Also, the approach works best if problems can be decomposed into independent sub-problems.

In past research, most distributed coordination approaches use auction and market techniques to divide tasks among robots, e.g. [7] and [8].

Although multi-agent systems are often used for multi-robot applications, the research mostly focusses on task allocation problems or on emergent behavior in large groups of small robots, called *swarms* [9].

IV. PROSA & DELEGATE MAS

In our approach, the environment becomes a *first class citizen* and a switch is made to an *open world assumption*. This results in a more scalable and flexible system in which resource awareness and allocation are explicit. The Product-Resource-Order-Staff Architecture (PROSA) is used to accomplish this. This *holonic* architecture defines four separate *holons*, depicted in Fig. 2 together with their interactions. A holon is an autonomous, self-reliant whole able to cope with disturbances, while also part of a larger whole.

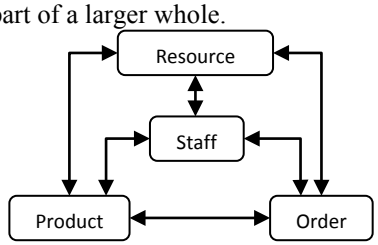

Fig. 2. Three basic PROSA holons and the optional staff holon are shown here. The arrows indicate their interactions. All agents have a counterpart in reality.

A. Structure

1) Product holon

A product holon encapsulates knowledge of a particular task type; how a task needs to be executed and what services or operations are required. These services and operations are offered by resource holons. This knowledge is captured in a process plan available to other holons. In its most simplistic form this plan contains a set of sequential steps required to assemble the product type it represents.

2) Order holon

An order holon represents a task *instance* that needs to be executed at a specific time instance. It is responsible to handle the required resource allocations to accomplish this. Therefore, the order holon consults its corresponding product holon to find out what services it needs and searches for the proper resources to accomplish it.

It's clear that each order holon requires a corresponding product holon of which it receives the process plan. In other words, for each desired task, a task description or specification is required.

Moreover, the order holon is responsible for the actual execution of the process plan and, thus, maintains also the state of the plan it received from its corresponding product holon.

Note that, due to interaction between order and product holon and the data encapsulation within holons, there is no *data model lock-in*. In other words, the representation of the same data, such as a process plan, might be completely different in order and product holons. Therefore, holons are not dependent on each other's internal data model.

In this application, an order represents a robot moving from one particular room to another at a given time instance. Order holons use delegate multi-agents systems (D-MAS) to delegate responsibilities to a set of lightweight agents. The behavior of these delegate MAS are based on food foraging of ants. In this context ants travel through resources querying for available allocation slots. A distinction is made between exploring ants and intention ants. The exploring ant searches for a possible solution for the order it is linked to. It requests resources when their execution will be finished and uses that time as the start time for the next resource request. Resources should be able to execute virtually in a what-if mode to answer the ant's request.

After a solution is found the ants sends a message to the Order agent with the result. The intention ant has a similar behavior as the exploring ant with the exception its task is to book the reservation on the resources rather than requesting a virtual execution.

3) Resource holon

A resource holon corresponds to any entity in the underlying world of interest offering services and operations to others, relevant for the application at hand. More concretely, any entity that will be used by other holons for those services, during some period of their lifetime, should be represented by their own resource holon. Some examples are robotic platforms, sensors in the building, doors, corridors and rooms.

Moreover, resource holons require explicit allocation and deallocation and also maintain specifications and capabilities of the underlying entity. They represent the limitations that other holons have to satisfy, e.g. how many holons are allowed to simultaneously allocate a particular resource holon and during which time intervals.

4) Staff holon

In view of the complexity of resource allocation, staff holons may assist other holons through advice on particular aspects. However, this advice is not binding. The use of staff holons is, thus, completely optional.

The addition of staff holons introduces a mechanism to overcome the problem that in some stable and predictable situations traditional centralized or hierarchical control architectures perform better.

The key difference between staff holons and traditional higher level decision making functionality found in hierarchical systems is the these holon's advisory role.

B. Why PROSA and D-MAS?

1) Complementary to existing approaches

It is extremely important to note that the paradigm presented in this dissertation has not the intention to replace existing state of the art. No, it offers an infrastructure in which state of the art of different robotics research domains can be integrated and reused.

Moreover, both on the software level as the hardware level integration and reuse are possible. E.g, robots running legacy software can be taken into account and thus integrated with robots employing the PROSA and D-MAS paradigm.

The main reason why the HMES or another PROSA and D-MAS implementation is complementary to exist-

ing state of the art is their level of operation or level of abstraction.

Indeed, the granularity level at which to adopt PROSA concepts, i.e. how fine grained to define PROSA holons, influences the reusability of previous developments.

For example, in a coarse grained approach a robotic wheelchair with obstacle avoidance software is represented as one resource holon capable of safe navigation and used as is, while in a fine grained approach the obstacle avoidance software is reimplemented by for instance dividing the environment in small resource holons and allocating these holons required to navigate from one location to the other.

2) Robustness and scalability through reflection of reality

In current robot systems, the physical entities required in an application, e.g. robot resources or objects in the environment are assumed available and no allocation is performed. Moreover, each component interacting with a particular entity has its own representation and maintains its own version of that entity.

Reflecting reality is one of the key features of the PROSA architecture. Combined with short-term forecasting, offered by D-MAS, this has lead in previous manufacturing applications to an increase in overall system robustness with respect to unexpected changes and disturbances.

Another benefit of reflection of reality and consequently the structural decomposition of these holons is scalability. In the multi-robot navigation scenario this can be defined with respect to the number of robots in the system as well as the size of the environment, e.g. the number of rooms and corridors.

In PROSA this multi-robot planning problem is split up and divided among all the environment resources, which each are responsible for their own part in reality. Adding a robot would then only influence the planning required in the relevant rooms and corridors this robot is travelling.

As mentioned before, the intention is not to replace the existing state of the art research in trajectory planning. Rather these state of the art techniques can be used locally in each environment resource, thus simplifying the global planning problem.

3) Preview control through bio-inspired D-MAS

By combining D-MAS and PROSA the coordination mechanism in the robot system evolves from reaction to prediction of disturbances and changes in the environment. Indeed, coupling resource allocation with D-MAS's short-term forecasting power, allows the system to anticipate and avoid bottlenecks or other navigational issues rather than have to deal with them.

This type of control is often referred to as *preview control* which, in contrast to feedforward or feedback control, deals with near future events which are estimated or forecast. In previous research preview control has mostly been applied in immediate robot control or shared control with a user. With PROSA and D-MAS this technique is also adopted to increase the robustness in the global planning.

V. Experiments and Results

A. Experimental setup

The experiments discussed in this paper were performed in a simulated environment consisting of a set of rooms connected by narrow corridors, wide enough for only one robot. Robots are represented by a rectangular shape and have a 360 degrees laser scanner with a one degree resolution. Each robot has the same path planner and path execution algorithm on board. When a robot receives a target it plans and starts executing a path to the target. At each time step, the path planner evaluates the currently followed path and recalculates if needed, thus avoiding dynamic obstacles appearing in the environment which were previously not detected.

Two scenarios were tested, corresponding to the environments shown in Fig. 1. For both scenarios 10 runs without and 10 runs with coordination were executed and data such as robot positions, navigation commands and elapsed time were logged for analysis. In order to interpret the gathered data, a set of benchmarks was defined.

B. Benchmarks

The benchmarks conducted in these experiments are not intended to be exhaustive.

The message here is that for each benchmark chosen by the designer of the experiment, the PROSA approach clarifies which set of holons require adaptation to optimize that benchmark. In our navigation experiments the resource holons representing the narrow corridors in the environment are adapted to allow only one robot at a time.

1) Excessive distance

Since the experiments involve multiple robots navigating from one point to another, a straightforward benchmark is to compare the travelled distance of a robot with the optimal distance, i.e. the originally calculated route between its start position and its target. The benchmark thus measures, in meters, the extra distance a robot had to travel to reach its goal. The longer this distance, the worse its path execution was. Both an absolute value in meters and a relative value in percentages are provided.

2) Elapsed time

Another measure that was chosen is the elapsed time navigating in the environment. Since only the time the robot is actually moving is taken into account, this benchmark also indicates how energy efficient the approach is. The shorter the time interval, the more efficient the robot reached its target. Alternatively the total time of the navigation task could be chosen as benchmark. This would also include waiting time in the coordinated approach due to allocations not granted yet.

3) Hausdorff distance

Besides the above two benchmarks, a more advanced method to compare the executed path to the original path was used, namely the Haussdorf distance. This distance calculates the maximum displacement between two paths with respect to the provided metric. In these experiments two metrics were calculated. A Euclidean metric, which

returns the shortest path from one trajectory to the other and a metric that measures rigid body motion, which also takes rotations into account. More detailed information on how these metrics are defined can be found in [10].

C. Without coordination

Fig. 3 and Fig. 4 show two common problems of a robocentric approach. The former shows a suboptimal robot trajectory. At the beginning of the run the corridor to the left was blocked by another robot, detected by this robot's laser scanner. Since no information about this blocking is provided, the robot plans a path around the obstacle through the rooms below. However, after some time, the robot which was blocking the corridor enters the room freeing the corridor. This triggers the path planner to change its route and calculate its path through the right corridor. All travelled distance to the south could have been avoided by waiting until the corridor was unblocked.

Fig. 4 shows a livelock situation which can occur if two robots enter the same corridor and no alternative route is possible due to a restricted environment. Again this problem could have been solved by ordering one of the two robots to wait until the other robot has exited the corridor. The more robots are navigating in an environment, the higher the chances for such a livelock are.

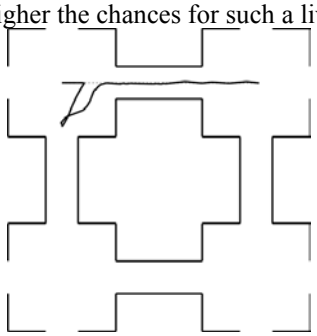

Fig. 3. The lack of information and intention awareness causes the robot to move down first and then heading back to take the corridor to the right.

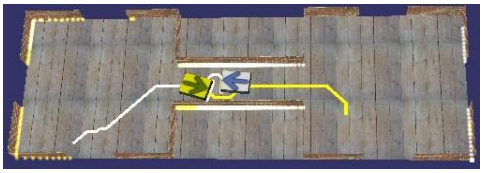

Fig. 4. Both robots enter the same corridor and fail to reach their target. This results in a livelock, i.e. continuously blocking each other in the corridor without finding a solution.

The results of the experiments without coordination are shown in Table I. In the environment with only two rooms and one corridor robots reached in 3 out 10 runs their target and in all three cases this was due to the fact that the time interval between their navigation order was high enough. This allowed the first robot to already exit the corridor before the second robot reached the entrance. In all the other cases the robots both entered the corridor causing a livelock. The elapsed time and excessive distance benchmarks shown in Table I only take successful runs into account, i.e. if the robots reached their target. If the target was not reached these benchmarks has little meaning, since elapsed time in a livelock would be infinite and excessive distance negative. Nevertheless, the Hausdorff distance can be interpreted, for runs not reaching the target, as a measure for the distance between the robot's end position and its target. Only taking the successful runs into account, this results in Hausdorff distances of 0.86 and 1.74 for respectively the Euclidean and rigid body motion metric.

TABLE I
BENCHMARKS FOR RUNS WITHOUT COORDINATION

Environment	2 by 1	2 by 2
Nr of Robots [#]	2	4
Nr of Runs [#]	10	10
Target Reached [%]	30%	100%
Absolute Excessive Distance [m]	1.060	1.142
Relative Excessive Distance [%]	10.40%	8.86%
Elapsed Time [s]	58.43	80.34
Hausdorff (Euclidean) [m]	3.813	1.795
Hausdorff (Rigid Body Motion) [m]	4.107	2.823

The experiments in the small environment with only two rooms and one corridor clearly show the need for coordination at bottlenecks, in this case a narrow corridor used by more than one robot in the environment. How this coordination is done, depends on the implementation of the corridor resource. In these experiments the resource scheduler only allowed one order at a time, thus allowing only one robot in the corridor at once. This scheduler could be replaced by a local multi-robot path planner allowing for instance that robots going the same direction can enter the corridor together. Since this implementation is separated from the other agents, such a change has little effect on the remainder of the system.

In the larger environment with four rooms, four corridors and four robots, all targets were reached since at any time if one robot blocks in a specific corridor, a solution can be found using the other corridor. However, this can lead to unwanted behavior, such as depicted in Fig 3, in which a robot first moves to one corridor, detects an obstacle and then moves to the other corridor.

Although in this larger environment the Hausdorff distance are smaller than those in the 2 by 1 environment, compared to only the successful runs of the smaller environment the difference is significant. In most runs one or more robots were *backtracking* and altering their path due to interference from others inside a corridor. This led to high Hausdorff distances.

C. With coordination

In the experiments with coordination through PROSA, robots are aware of each other's intentions and only enter a corridor after the scheduler of that particular corridor resource has given them a valid slot. This way livelock or backtracking is avoided. Whenever a robot is not able to enter a corridor yet, it waits in a transit zone in front of the corridor's entrance. These transit zones are a chosen in such a way that robots already in the corridors are not blocked by waiting robots during their exit from the corridor. Since the robot has allocated this corridor resource before starting to navigate through it, it is certain its original path can be executed. In the room resource the robot deviates from its original path to avoid a collision with the other robot waiting in the transit zone. This

TABLE II
BENCHMARKS FOR RUNS WITH COORDINATION

Environment	2 by 1	2 by 2
Nr of Robots [#]	2	4
Nr of Runs [#]	10	10
Target Reached [%]	100%	100%
Absolute Excessive Distance [m]	0.098	0.315
Relative Excessive Distance [%]	0.91%	2.63%
Elapsed Time [s]	66.03	83.42
Hausdorff (Euclidean) [m]	0.239	0.383
Hausdorff (Rigid Body Motion) [m]	1.340	1.514

deviation can be reduced by making this transit zone larger or moving it further away from the doorway.

In Fig. 5 a trajectory of a robot using coordination in the 2 by 2 environment is shown. The robot navigates from transit zone to transit zone until it reaches its target. Although such a path is suboptimal in distance, the execution is guaranteed to be smooth since resource allocation explicitly grants the robot passage.

The results of these experiments with coordination are shown in Table II. With properly chosen transit zones, a significant decrease of excessive distance is achieved, both in absolute and relative values. The higher elapsed time compared to the reactive approach is largely due to the slightly longer trajectories and the intermediate stops at the transit zones causing more low speed travel. Also Hausdorff distances are significantly lower than the approach without coordination. Performing a Wilcoxon rank-sum test on the excessive distance data results in a z-score of -4.205035 and p-value of 0.00001. This z-score is lower than the z-score of the one-tailed significance level of 5%, -1.64485, thus the null hypothesis $\mu_{coordinated} > \mu_{reactive}$ can be rejected. Analogously, hypothesis tests on the Hausdorff distances with Euclidean metric and rigid body motion metric results in the rejection of the null hypotheses, with respectively z-scores of -3.06958 and -2.75204 and p-values of 0.00107 and 0.00296, at the same significance level of 5%. It can also be shown that although the mean execution time is higher in the coordinated approach the difference is not significant, with z-score 0.19245 and p-value 0.42370.

VI. CONCLUSIONS AND FUTURE WORKS

The experiments discussed in this paper illustrated the need for coordination in multi-robot navigation and the benefits of resource allocation.

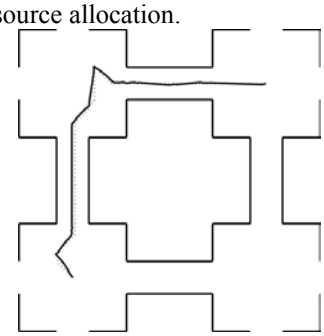

Fig. 5. A trajectory of robot moving from one room to the other using PROSA's coordination mechanism. The deviations indicate either movements to transition zones or avoiding other robots in its path.

Explicitly representing the environment as resource holons, enhances system flexibility and scalability. The planning in a multi-robot application is distributed to these resources responsible for the planning, i.e. allocation, in the physical entity they represent. Also, restrictions on the allocation of particular resources, such as narrow corridors, are applied on these specific components without the need to change other components.

Experimental data in simulation showed a coordinated approach with explicit resource allocation achieves better performance in path execution, measures by the defined benchmarks. It should be noted that these experiments only illustrate the applicability of the approach and not limit it to this specific choice of performance criteria.

Future work will include experiments on real hardware, i.e. a heterogeneous fleet of robots, illustrating the system flexibility. Distributing the planning over several environment resources lowers the computational complexity and suggests an improved scalability, but validation is still required.

Additional material on these experiments, such as videos of runs with and without coordination, can be found online at http://people.mech.kuleuven.be/~jphilips/coordination/.

REFERENCES

[1] P. Valckenaers and H. Van Brussel, "Holonic Manufacturing Execution Systems", CIRP Annals - Manufacturing Technology, vol. 54, nr. 1, pp. 427-432, 2005.

[2] J. Van Belle, J. Philips, O. Ali, B. Saint Germain, H. Van Brussel and P. Valckenaers, "A Service-Oriented Approach for Holonic Manufacturing Control and Beyond", Service Orientation in Holonic and Multi-Agent Manufacturing Control, 2012.

[3] M. Peasgood, C. Clark and J. McPhee, "A Complete and Scalable Strategy for Coordinating Multiple Robots Within Roadmaps", IEEE Transactions on Robotics 24.2, pp. 283 –292, 2008.

[4] J. van den Berg, J. Snoeyink, M. Lin, and D. Manocha, "Centralized path planning for multiple robots: Optimal decoupling into sequential plans", Proceedings of Robotics: Science and Systems, 2009.

[5] G. Wagner and H. Choset, "M*: A complete multirobot path planning algorithm with performance bounds", Intelligent Robots and Systems, 2011.

[6] B. Brummit, and A. Stentz, "GRAMMPS: A Generalized Mission Planner for Multiple Mobile Robots", Proceedings of the IEEE International Conference on Robotics and Automation, pp. 1564–1571, 1998.

[7] B. Gerkey and M. Mataric, "Sold!: auction methods for multirobot coordination", IEEE Transactions on Robotics and Automation, 18.5, pp. 758–768, 2002.

[8] M. Dias, A. Stentz, "A free market architecture for distributed control of a multirobot system", 6th International Conference on Intelligent Autonomous Systems, pp. 115–122, 2000.

[9] M. Dorigo and T. Stützle, "Ant Colony Optimization", Scituate, MA, USA, Bradford Company, 2004.

[10] A. Hüntemann, "Probabilistic Human-Robot Navigation – Plan Recognition, User Modelling and Shared Control for Robotic Wheelchairs", PhD Thesis, 2011.

[i] The paper reflects the work funded by the Research Council of the KU Leuven - concerted research action GOA-ACDPS (Autonomic Computing in Decentralised Production Systems) and European project BRICS (FP7-231940, Best Practice in Robotics). Johan Philips is with Department of Mechanical Engineering, Katholieke Universiteit Leuven, B-3001 Heverlee, Belgium.
Contact address: johan.philips@mech.kuleuven.be